More on Branching in MIPS
More on Memory in MIPS

CS 64: Computer Organization and Design Logic
Lecture #8

Ziad Matni
Dept. of Computer Science, UCSB
Administrative

• Adjusted (pushed out) due dates for many lab assignments

• Assignment #4: Due Fri. 2/10

• Midterm results will be released on Th. 2/9
Lecture Outline

• More branches in MIPS
  – With examples for nested branches

• Memory in MIPS
  – With examples for memory access and array vars
More Branches in MIPS

Let’s take a look at:

• else_if.asm
• nested_if.asm
• nested_else_if.asm
else_if.asm

int y
if (x == 5) {
  y = 8
}
else if (x < 7) {
  y = x + x
}
else {
  y = -1
}
print(y)
if (x < 6) {
    if (x > 1) {
        print("x is in (1,6)\n")
    }
    else {
        print("x <= 1\n")
    }
} else {
    print("x >= 6\n")
}
nested_else_if.asm

```asm
if (x < 11) {
    if (x > 5) {
        str = "(5, 11)\n"
    }
    else if (x == 0) {
        str = "0\n"
    }
    else {
        str = "< 11\n"
    }
}
else {
    str = "x >= 11\n"
}
print(str)
```
RECALL THAT:

- Two base instructions:
  - load-word (lw) from memory to registers
  - store-word (sw) from registers to memory

- MIPS lacks instructions that do more with memory than access it (e.g., retrieve something from memory and then add)
  - Operations are done step-by-step
  - Mark of RISC architecture
Global Variables

Recall:

• Typically, global variables are placed directly in memory, not registers

• Let’s take a look at: access_global.asm
Load Address (la) and Load Word (lw)

.data
myVariable: .word 42

main:
la $t0, myVariable  \leftarrow \text{WHAT'S IN $t0??}
lw $t1, 0($t0)  \leftarrow \text{WHAT DID WE DO HERE??}

li $v0, 1
move $a0, $t1
syscall  \leftarrow \text{WHAT SHOULD WE SEE HERE??}
access_global.asm

Store Word (sw)

li $t1, 5
sw $t1, 0($t0)  \(\Leftarrow \) WHAT’S IN $t0 AGAIN??

li $t1, 0
lw $t1, 0($t0)  \(\Leftarrow \) WHAT DID WE DO HERE??

li $v0, 1
move $a0, $t1
syscall  \(\Leftarrow \) WHAT SHOULD WE SEE HERE??
Arrays

• Question:
As far as memory is concerned, what is the *major difference* between an *array* and a *global variable*?
  – Arrays contain multiple elements

• Let’s take a look at:
  • print_array1.asm
  • print_array2.asm
  • print_array3.asm
int myArray[]
    = {5, 32, 87, 95, 286, 386}
int myArrayLength = 6

int x

for (x = 0; x < myArrayLength; x++) {
    print(myArray[x])
    print("\n")
}
int myArray[] = {5, 32, 87, 95, 286, 386}
int myArrayLength = 6
int* p

for ( p = myArray; p < myArray + myArrayLength; p++) {
    print(*p)
    print("\n")
}
YOUR TO-DOs

• Assignment #4: Due Fr. 2/10